

Introduction to the Exchange Management Shell

The Microsoft Exchange Management Shell, built on Microsoft Windows PowerShell technology, is a powerful management interface that you can use to manage every aspect of Exchange Server 2007 from the command line. You can interact directly with the Exchange Management Shell via the command line, write scripts for automation, or integrate your applications with the underlying engine. Simply put, the Exchange Management Shell puts you in control of your Exchange infrastructure. Here are some of its key benefits:

- **Scripting made safe and easy** The Exchange Management Console doesn't just use the Exchange Management Shell to perform operations; the wizards show you what Exchange Management Shell commands they're performing. Copy the commands the wizards create and paste them directly onto the command line or into a script. After you've modified the parameters to your liking, test your scripts or commands by using the *WhatIf* parameter, which lets you see the results before you make any changes to your environment.
- **Maximize your time, not your frustration** The Exchange Management Shell gives you more power than any other scripting solution to tackle the problems you face. It gives you a robust and flexible scripting platform to integrate with your current scripting solutions, and it lets you communicate natively with data sources like COM and Windows Management Instrumentation (WMI). Use your time wisely, creating useful solutions instead of creating the framework to support them.
- **Repeat after us: No more text parsing!** Based on Microsoft .NET, the Exchange Management Shell cmdlets accept and return structured data instead of plain text. This enables you to easily pass data back and forth without messy text parsing. Turn even CSV and XML files into objects that you can manipulate and consume on a single line.
- **Clear and simple** All cmdlets are named by using verb-noun pairs that clearly identify the feature or component being managed, and the action that's being taken. You don't have to guess what a cmdlet does anymore. When you see the **Move—Mailbox** cmdlet, you know exactly what it's used for.

The following table gives you a quick overview of the Exchange Management Shell fundamentals. Most cmdlets use these basic features to enable you to easily administer your Exchange organization.

Common Cmdlet Actions	
New-	The New verb creates a new instance of something, such as a new configuration setting, a new database, or a new SMTP connector.
Remove-	The Remove verb removes an instance of something, such as a mailbox or transport rule. All Remove- cmdlets support the <i>WhatIf</i> and <i>Confirm</i> parameters. For more information about these parameters, see the "Important Parameters" section.
Enable-	The Enable verb enables a setting or mail-enables a recipient.
Disable-	The Disable verb disables an enabled setting or disables a mail-enabled recipient. All Disable- cmdlets also support the <i>WhatIf</i> and <i>Confirm</i> parameters. For more information about these parameters, see the "Important Parameters" section.
Set-	The Set verb modifies specific settings of an object, such as changing the alias of a contact or changing the deleted item retention value on a mailbox database.
Get-	The Get verb queries a specific object or a subset of a type of object, such as a specified mailbox, all mailbox users, or mailbox users under a domain.
Important Parameters	
-Identity	The <i>Identity</i> parameter identifies the unique object for the cmdlet and is generally used with Enable , Disable , Remove , Set , and Get cmdlets. The <i>Identity</i> parameter is also a positional parameter, which means that you don't have to specify <i>Identity</i> when you specify the parameter's value on the command line.
-WhatIf	The <i>WhatIf</i> parameter instructs the cmdlet to simulate the actions that the cmdlet would take on the object. By using the <i>WhatIf</i> parameter, you can "test-drive" a command before you make any real changes to your environment.
-Confirm	The <i>Confirm</i> parameter causes the cmdlet to pause and requires the administrator to acknowledge what the cmdlet will do before processing continues.
-Validate	The <i>Validate</i> parameter, which is available on certain cmdlets, causes the cmdlet to check that all prerequisites for running the operation are satisfied and that the operation will complete successfully.
Tips & Tricks	
Get-Command <i>*keyword*</i>	Retrieves cmdlets that have <i>keyword</i> in the cmdlet
Get-cmdlet Format-List	Displays the output of the query in a formatted list
Help <i>cmdlet</i>	Retrieves help information for any cmdlet in Exchange Server 2007 (Example: Help Get-Mailbox)
Help <i>cmdlet</i> - <Detailed Full Examples>	Retrieves additional help information for any cmdlet in Exchange 2007. (Example: Help New-JournalRule -Detailed)
Help <i>cmdlet</i> -Parameter <i>parameter</i>	Retrieves help for a specific parameter or multiple parameters on a specific cmdlet. To retrieve multiple parameters that contain the same string, enclose the string with wildcard characters. (Example: Help Set-Mailbox -Parameter *email*)
Help -Role <i>*server role*</i>	Retrieves a list of all cmdlets that manage the components of a specific server role. (Example: Help -Role *Mailbox*) For information about valid values, see "Getting Help" in the Exchange 2007 Help file.
Help -Component <i>*component*</i>	Retrieves a list of all cmdlets that manage a specific feature. For information about valid values, see "Getting Help" in the Exchange 2007 Help file.
<i>cmdlet</i> <tab> <i>cmdlet -parameter</i> <tab>	Use tab completion to complete a cmdlet name or parameter name. Enter a partial name for <i>cmdlet</i> , and then press the TAB key to cycle through all cmdlets that contain the text you specified. Or if you want to cycle through all the parameters on a cmdlet, specify the cmdlet and a partial parameter name, and then press the TAB key to cycle through all the available parameters. You can also use wildcard characters, for example, <i>Help *UM*</i> .
Get-MailboxDatabase <i>Store1</i> Get-Mailbox	Pipelining enables you to pass objects from one cmdlet to another. All verbs in a noun-set (cmdlets with the same noun) can pass objects between each other by using pipelines, and many cmdlets outside the same noun-set can do the same. The example at left returns a list of all mailboxes located in <i>Store1</i> .

Introduction to the Exchange Management Shell

You can get help directly from the command line.

Multi-thread support lets you perform multiple tasks faster and more efficiently.

```

Machine: mbx CWD: C:\
[PS] C:\>Get-Help Move-Mailbox
[PS] C:\>

Jessica_Arnold
Moving messages.
[Progress bar]
Jeff_Phillips
Moving messages. Calendar (1/2)
[Progress bar]

[PS] C:\>
[PS] C:\>Get-MailboxDatabase -Status Research | Format-Table Name, Mount*

Name                MountAtStartup      Mounted
----                -
Research            True                True

[PS] C:\>
[PS] C:\>Get-MailboxDatabase Sales | Get-Mailbox

Name                Alias                ServerName            ProhibitSendQuota
----                -
Jessica_Arnold      Jessica_Arnold        win2003ns             unlimited
Jeff_Phillips       Jeff_Phillips         win2003ns             unlimited

[PS] C:\>Get-MailboxDatabase Sales | Get-Mailbox | Move-Mailbox -TargetDatabase
Research
    
```

View the progress of longer running commands as they perform their tasks.

View only the information you want to see in the format you want. Use wildcards to match multiple properties or if you don't remember the exact name of the property.

Use pipelines to pass objects between cmdlets so that you can retrieve specific objects only, such as all the mailboxes that are located in a specific mailbox database.

With native object support, you can instantly act on the data that is returned to you. For example, you can move all the mailboxes in one mailbox store to another mailbox store.

```

Machine: Win2003gw CWD: C:\
[PS] C:\>Help Get-AddressRewriteEntry -Detailed

NAME
Get-AddressRewriteEntry

SYNOPSIS
Use the Get-AddressRewriteEntry cmdlet to view an existing address rewrite
entry that rewrites sender and recipient e-mail addresses in e-mail message
s that are sent to and from an e-mail organization.

SYNTAX
get-addressrewriteentry [-Identity <AddressRewriteEntryIdParameter>] [-Doma
inController <Fqdn>] [<CommonParameters>]

DETAILED DESCRIPTION
In Microsoft Exchange Server 2007, address rewriting lets you modify the ad
resses of senders and recipients on messages that enter and leave an Excha
nge 2007 organization. You configure Address Rewriting agents on the Receiv
e connector and Send connector on a computer that has the Edge Transport se
rver role installed.
To run the Get-AddressRewriteEntry cmdlet on a computer that has the Edge T
ransport server role installed, you must log on by using an account that is
a member of the local Administrators group on that computer.

PARAMETERS
-DomainController <Fqdn>
To specify the fully qualified domain name (FQDN) of the domain control
ler that writes this configuration change to the Active Directory direc
tory service, include the DomainController parameter on the command. Th
e DomainController parameter is not supported on computers that have th
e Edge Transport server role installed. The Edge Transport server role
writes only to the local ADAM instance.

-Identity <AddressRewriteEntryIdParameter>
This parameter specifies the address rewrite entry to be retrieved. The
Identity parameter will accept a GUID, or the unique address rewrite n
ame. You can omit the Identity parameter label.

<CommonParameters>
This cmdlet supports the common parameters: -Verbose, -Debug,
-ErrorAction, -ErrorVariable, and -OutVariable. For more information,
type, "get-help about_commonparameters".

Get-AddressRewriteEntry
Get-AddressRewriteEntry "Address rewrite entry for contoso.com" | Format-Li
st

REMARKS
For more information, type: "get-help Get-AddressRewriteEntry -detailed".
For technical information, type: "get-help Get-AddressRewriteEntry -full".
    
```

Cmdlet name.

Quick introduction to the cmdlet and what it does.

Lists every parameter available, and in what combinations they can be used.

A detailed description of the cmdlet, how it interacts with Exchange, and what permissions are required to run it.

A description of each available parameter and the input values they take.

Real-world examples of commands that you're likely to perform.